VMSub : Uma ferramenta de Simulação do Processo de Substituição de Páginas em Gerência de Memória Virtual

Hugaleno C. Bezerra[1], **Stéphanie A. Braga**[1], **Fernando P. Garcia**[1]

[1]Departamento de Telemática – Intituto Federal do Ceará (IFCE)
Av. 13 de maio, 2081, 60040-531 – Fortaleza – CE – Brasil

{hugaleno,stephanie.abraga}@gmail.com, fernandoparente@ifce.edu.br

Resumo. *Este artigo apresenta um software para auxílio no ensino de disciplinas que abordam o conteúdo de gerência de memória virtual. São implementados oito algoritmos de substituição de página, incluindo o Ótimo para fins de comparação de desempenho. A ferramenta possui um modo de simulação massiva dos dados, para fins de comparação de desempenho dos algoritmos e um modo interativo, onde o usuário pode visualizar o "passo-a-passo"de cada etapa do processo de substituição de páginas.*

Palavras Chave *- memória virtual, substituição de páginas, simulação, ferramenta de aprendizagem*

VMSub : A tool for simulation of pages substitution process virtual memory management

Abstract. *This paper presents a software of support in the teaching of disciplines that approach the content of virtual memory management. Eight algorithms of page replacement have been implemented, including the optimal one for performance comparison purposes. The tool has a massive simulation mode of data for the purpose of dis performance comparison algorithms and an interactive mode, where the user can view the "step by step"of each stage of the page replacement process.*

Keywords *- virtual memory, page substitution, simulation, learning tool*

1. Introdução

A disciplina de sistemas operacionais, presente em cursos na área de Tecnologias de Informação e Comunicação (TICs), tem como objetivo fornecer ao estudante conhecimentos relativos aos fundamentos da arquitetura de um sistema operacional, tais como gerência do processador, de memória e de dispositivos de entrada e saída utilizados pelo computador. Entender esses pontos, no entanto, não é uma tarefa fácil.

Em geral, nos cursos da área de Computação, os alunos de disciplinas de Sistemas Operacionais e afins apresentam dificuldades para entender a interação entre hardware e software que envolve o processo de gerência de memória, mais precisamente a administração e funcionamento da memória virtual. Um dos possíveis fatores é o alto grau de abstração requerido por este tema, bem como o elevado nível de conhecimento acerca do funcionamento dos elementos envolvidos. Experiências observadas entre professores e alunos mostram como é difícil a compreensão dos assuntos dessa disciplina, bem como a aplicação prática dos mesmos (MAIA; MACHADO, 2004).

Segundo (FONSECA; NASCIMENTO, 2009), a forma tradicional que a disciplina é ensinada, utilizando apenas teoria, dificulta o entendimento do aluno e, em alguns casos, pode até desmotivá-lo, por conta da falta de ferramentas capazes de mostrar na prática os conceitos vistos na teoria. O uso de uma ferramenta que mostre o funcionamento desses algoritmos na prática pode auxiliar o aluno a compreender a teoria estudada.

Neste artigo apresentamos uma ferramenta, denominada VMSub, que mostra o funcionamento de oito algoritmos de substituição de páginas, com entrada de teste com até um milhão de referências. Os resultados obtidos são apresentados em um gráfico e o aluno pode então observar o funcionamento de cada algoritmo, facilitando assim o seu aprendizado.

O presente trabalho está organizado da seguinte forma: A Seção 2 explica o conceito de memória virtual e o processo de paginação. A Seção 3 aborda o processo de substituição de páginas e apresenta oito algoritmos de substituição de páginas, exemplificados por um *running example*. Na Seção 4 é apresentada a aplicação desenvolvida. Na Seção 5 são apresentados os resultados obtidos com a aplicação. a Seção 6 é feito um comparativo do VMSub com outros trabalhos. Finalmente, na Seção 7 é apresentada a conclusão deste artigo.

2. Memória Virtual

A memória virtual é uma técnica que permite a execução de processos que podem não estar inteiramente na memória principal. A principal vantagem visível desse esquema é que os programas podem ser maiores do que a memória física. Além disso, ele abstrai a memória principal em um vetor extremamente grande e uniforme de armazenamento, separando a memória lógica, conforme vista pelo usuário, da memória física (SÍLBERSCHATZ; GALVIN; GAGNE, 2001).

Nos primeiros sistemas a implementar estratégias de memória virtual, processos inteiros eram transferidos da memória para o disco rígido e vice-versa. Esse procedimento, denominado troca (*swapping*), permite liberar grandes áreas de memória a cada transferência, e se justifica no caso de um armazenamento com tempo de acesso muito elevado, como os antigos discos rígidos. Cada programa tem seu próprio espaço de endereçamento, que é dividido em blocos chamados páginas. Cada página é uma série contígua de endereços. Essas páginas são mapeadas na memória física, mas nem todas precisam estar, simultaneamente, na memória física para executar o programa (TANENBAUM, 2010; MAZIERO, 2014).

2.1. Falta de página (*Page fault*)

Quando um processo acessa uma página, a MMU (*Memory Management Unit* - Unidade de Gerenciamento de Memória) verifica se a mesma está mapeada na memória RAM (*Random Access Memory*) e, em caso positivo, faz o acesso ao endereço físico correspondente. Caso contrário, a MMU gera uma interrupção de falta de página (*page fault*) que força o desvio da execução para o sistema operacional. Nesse instante, o sistema deve verificar se a página solicitada não existe ou se foi transferida para o disco. Caso a página solicitada tenha sido transferida para o disco, o processo deve ser suspenso enquanto o sistema transfere a página de volta para a memória RAM e faz os ajustes necessários na tabela de páginas. Uma vez a página carregada em memória, o processo pode continuar sua execução (MAZIERO, 2014).

3. Substituição de páginas

Quando ocorre uma falta de página, o sistema operacional precisa escolher uma página a ser removida da memória física a fim de liberar espaço para uma nova página a ser trazida para a memória física. Embora seja possível escolher aleatoriamente uma página a ser descartada a cada falta de página, o desempenho do sistema será melhor se a página escolhida for uma que não estiver sendo muito usada. A escolha correta das páginas a remover da memória física é um fator essencial para a eficiência do mecanismo de memória virtual (TANENBAUM, 2010; MAZIERO, 2014).

Nesta seção são mostrados os oito algoritmos implementados na aplicação através de um *Running example*, Tabela 1, onde cada algoritmo é analisado separadamente para escolha da página a ser substituída.

Espaço de memória						
Posições:		**4**	**3**	**2**	**1**	**0**
Páginas:	Número da página	5	6	7	2	1
	Tipo de Acesso	Leitura	Escrita	Leitura	Escrita	Leitura
	Bit de Referência	1	1	1	1	1
	Bit de Modificação	0	1	0	1	0
	Frequência de Acesso	1	5	9	2	4

Tabela 1. ***Running example***

A Tabela 1 ilustra um espaço de memória física (RAM) com cinco posições, todas previamente preenchidas. Assume-se que a ordem de carregamento das páginas dá-se da direita para esquerda, sendo a página da posição 0 a primeira a ser carregada e a da posição 4, a última. Cada página possui cinco atributos: Número, Tipo de Acesso, Bit de Referência, Bit de Modificação e Frequência de Acesso.

O Tipo de Acesso informa se a operação feita sobre a página é de apenas leitura ou uma operação de escrita. O Bit de Referência é um atributo de cada página que assume os valores 1 e 0. Sempre que a página é referenciada, esse bit é alterado para 1. O bit é alterado para 0 quando o processador zera os Bits de Referência de todas as páginas, em intervalos de tempo variáveis de acordo com o sistema operacional, ou quando o próprio algoritmo de substituição utilizado altera essa valor. Já o Bit de Modificação também é um atributo que assume valores 1 e 0, porém o seu valor só é alterado para 1 quando há uma operação de escrita na página. Se for uma operação de leitura, este último permanece 0. A Frequência de Acesso refere-se a quantas vezes a página foi acessada após ter sido carregada na memória. Este contador é incrementado a cada nova referência a esta mesma página.

Os algoritmos analisados neste artigo são: FIFO, Segunda Chance, LFU, MFU, *Rand*, LRU, NRU e Ótimo.

FIFO *(First In First Out)*: A ideia central deste algoritmo é que as páginas que estão a mais tempo na memória podem ser substituídas, ou seja, as primeiras que entram são as primeiras que saem (JÚNIOR, 2004). Nesse exemplo, a página a ser removida é a 1 na posição 0 da memória, já que foi a primeira a ser carregada.

Segunda Chance: Uma melhoria simples do FIFO consiste em analisar o bit de referência de cada página candidata para saber se ela foi acessada recentemente. Caso tenha sido, essa página recebe uma ”segunda chance”, voltando para o fim da fila com

seu bit de referência ajustado para zero (MAZIERO, 2014). Nesse exemplo, a página 1 da posição 0 não seria removida, como no FIFO, pois apresenta bit de referência R com valor 1, então esta página seria inserida no início da fila com seu bit R zerado e a página 2 da posição 1 seria a candidata a remoção.

LFU (*Least Frequently Used*): Neste algoritmo a página com o menor contador, menos referenciada, será substituída. Este algoritmo leva em consideração que se uma página, até o momento da substituição, foi menos referenciada então ela deve ser removida (SÍLBERSCHATZ; GALVIN; GAGNE, 2001). No exemplo, a candidata para remoção seria a página 5 da posição 4, pois possui a menor Frequência de Acesso, o que indica que ela foi usada menos vezes.

MFU (*Most Frequently Used*): Este algoritmo baseia-se no argumento de que a página com menor contagem provavelmente acaba de chegar à memória e ainda deverá ser utilizada (SÍLBERSCHATZ; GALVIN; GAGNE, 2001). Nesse caso, a candidata para remoção seria a página 7 da posição 2, pois possui o maior contador, o que indica que ela foi usada mais vezes.

Rand: Este algoritmo escolhe uma página de forma aleatória para descarte, eliminando o custo computacional de escolha da página a ser removida. Sob essa estratégia, cada página da memória principal tem a mesma probabilidade de ser selecionada para substituição (DEITEL; DEITEL; CHOFFNES, 2005). Nesse exemplo, qualquer uma das páginas é candidata a remoção.

LRU (*Least Recently Used*): Neste algoritmo quando ocorre uma falta de página, a página a ser eliminada será a menos recentemente utilizada. Embora o LRU seja realizável, possui um maior custo computacional, pois exige que a fila de páginas seja ordenada a cada nova referência. Nesse exemplo, de acordo com a ordem de carregamento das páginas para memória previamente informada, a página da posição 4 seria a candidata a remoção, já que é a página menos recentemente carregada na memória.

NRU (*Not Recently Used*): Como citado anteriormente, o algoritmo Segunda Chance, leva em consideração apenas o bit de referência de cada página a ser substituída. Já o algoritmo NRU melhora essa escolha, ao considerar também o bit de modificação, que indica se o conteúdo de uma página foi modificado após ela ter sido carregada na memória (MAZIERO, 2014). Já que se tem 2 bits a serem analisados, tem-se 4 possibilidades divididas nas seguintes classes: 3. Referenciada, modificada 2. Referenciada, não modificada 1. Não referenciada, modificada 0. Não referenciada, não modificada

A candidata a remoção é a que tiver a menor classe. No caso de páginas com a mesma classe, será utilizado o critério de FIFO para a escolha entre elas. Nesse exemplo, há 3 páginas com a classe 2, então, pelo critério FIFO, a página a ser removida é a 7 da posição 2, pois foi referenciada, R igual 1, mas não foi modificada, M=0 e foi a primeira a ser carregada.

Ótimo: Idealmente, a melhor página a remover da memória em um dado instante é aquela que ficará mais tempo sem ser usada pelos processos. Esta ideia simples define o algoritmo ótimo. Entretanto, como o comportamento futuro dos processos não pode ser previsto com precisão, este algoritmo não é implementável. Mesmo assim ele é importante, porque define um limite máximo conceitual. Assim, seu resultado serve como parâmetro para a avaliação dos demais algoritmos (MAZIERO, 2014).

4. O VMSub

A ideia de desenvolver uma aplicação surgiu durante a disciplina de Sistemas Operacionais do curso de Engenharia de Telecomunicações do IFCE (Instituto Federal de Educação, Ciência e Tecnologia do Ceará), quando foram estudados os algoritmos de substituição de páginas. Desde então, a aplicação começou a ser desenvolvida na linguagem de programação Java, devido às ferramentas e documentações já disponibilizadas pela sua API (*Application Programming Interface*).

A aplicação tem o objetivo de simular os algoritmos de substituição de páginas de uma forma interativa, visando obter os resultados da simulação desses algoritmos e facilitar o aprendizado dos mesmos. A Figura 1 mostra um diagrama que ilustra o funcionamento e características gerais da aplicação.

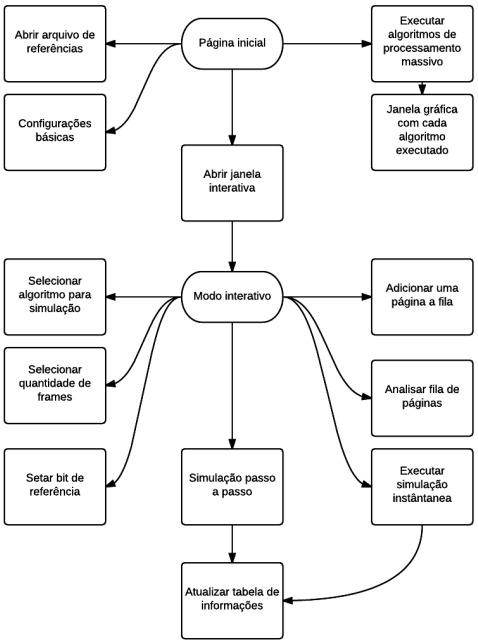


Figura 1. Diagrama de Funcionalidades

Para tornar a aplicação mais didática foram implementadas dois modos de funcionamento:1- Análise de Desempenho, que faz o processamento massivo dos dados e permite a análise de desempenho de cada algoritmo e 2- Modo interativo, onde o usuário pode acompanhar a execução de cada algoritmo, observando cada ação realizada durante o processo de substituição de páginas. Abaixo está descrito como cada uma das opções funcionam.

4.1. Modo de Análise de Desempenho

Esse modo está na própria tela inicial da aplicação e, antes de iniciar o processo de simulação, o usuário precisa fazer três configurações: carregar um arquivo que representa as páginas a serem referenciadas, estabelecer o intervalo para a quantidade de *frames* da memória e o intervalo para zerar o Bit de Referência das páginas. Esta última configuração é necessária para simular um dos procedimentos do sistema operacional, onde é zerado o Bit de Referência de todas as páginas carregadas na memória. Na Figura 2 é mostrada a tela inicial da aplicação.

O arquivo a ser carregado deve estar no seguinte formato: 5W-7R-8W-2R-5W-7W, com quantas referencias o usuário desejar. A numeração representa o número de cada

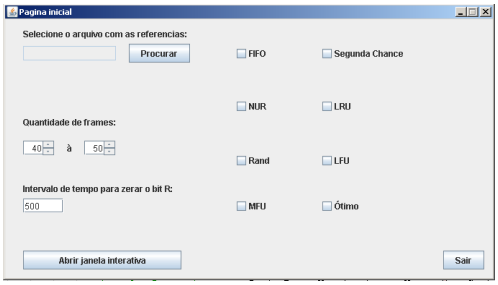
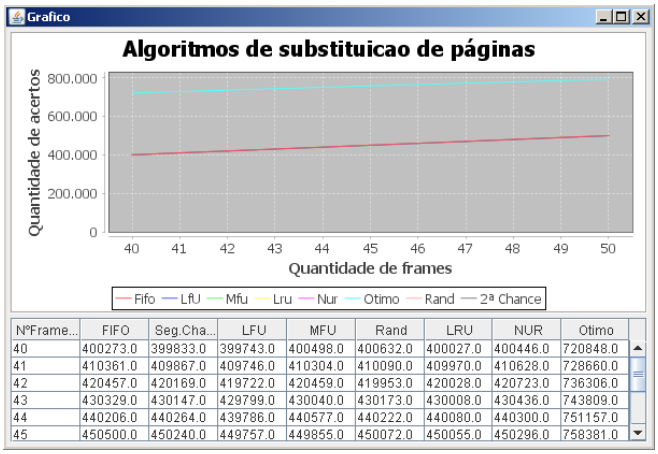


Figura 2. Tela Inicial

página e o caractere que segue a numeração representa o tipo de acesso, R para leitura e W para escrita. Os dados das referências de páginas contidas no arquivo não são de um programa real. Na tela, o intervalo de *frames* representa uma memória com diferentes quantidades de *frames*. No exemplo da Figura 2 é feita a simulação dos algoritmos onze vezes, com 40 *frames*, 41, até 50 *frames*. O intervalo para zerar o Bit de Referência das páginas é indicado pelo número de páginas que já foram carregadas na memória. No exemplo, o arquivo continha 1000000 (um milhão) de páginas e o Bit R de todas as páginas era zerado a cada 500 páginas referenciadas.

Após feitas essas configurações, o usuário pode marcar as opções dos algoritmos que deseja simular. Uma janela contendo os resultados das simulações é gerada automaticamente, Figura 3.



NºFrame...	FIFO	Seg.Cha...	LFU	MFU	Rand	LRU	NUR	Otimo
40	400273.0	399833.0	399743.0	400498.0	400632.0	400027.0	400446.0	720848.0
41	410361.0	409867.0	409746.0	410304.0	410090.0	409970.0	410628.0	728660.0
42	420457.0	420169.0	419722.0	420459.0	419953.0	420028.0	420723.0	736306.0
43	430329.0	430147.0	429799.0	430040.0	430173.0	430008.0	430436.0	743809.0
44	440206.0	440264.0	439786.0	440577.0	440222.0	440080.0	440300.0	751157.0
45	450500.0	450240.0	449757.0	449855.0	450072.0	450055.0	450296.0	758381.0

Figura 3. Gráfico comparativo

Os dados obtidos de cada algoritmo são atualizados em um gráfico e em uma tabela, onde é possível ver a relação do número de acertos para todas as quantidade de *frames* do intervalo estabelecido. O gráfico possibilita a comparação entre o desempenho de cada algoritmo com o algoritmo ótimo (melhor algoritmo possível, porém não realizável). Na Figura 3, a linha azul representa o número de acertos do algoritmo ótimo e as outras representam o número de acertos dos outros algoritmos. Como o número de acertos para os algoritmos são parecidos, as linhas praticamente se sobrepõem, ficando em evidência apenas a linha vermelha na figura. Para visualizar melhor, o usuário pode dar um *zoom* no gráfico.

A tabela mostra os números de acertos e é possível ver, de forma mais clara, essa diferença na quantidade de acertos dos algoritmos realizáveis com a quantidade de acertos do algoritmo ótimo, mostrando a superioridade expressiva deste. Em relação aos outros,

podemos notar que, apesar de usarem técnicas diferentes para escolha da página a ser substituída, apresentam desempenho, quanto ao número de acertos, muito próximos. Pela tabela pode-se ver que a medida que o número de *frames* da memória principal aumenta, aumenta-se também o número de acertos de cada algoritmo, o que é esperado, pois quanto mais páginas puderem ser carregadas na memória principal, menor será a necessidade de troca de páginas pelo sistemas operacional.

Para a implementação desse modo foi usado o recurso de *threads*, oferecido pela linguagem Java para executar cada procedimento em uma *thread* separada, aumentando, dessa forma, o aproveitamento dos recursos da máquina e diminuindo o tempo de espera do usuário.

4.2. Modo Interativo

Na tela principal há também um botão que, ao ser acionado, carrega a interface para o Modo Interativo, mostrada na Figura 4. Nesse modo, o usuário também precisa realizar as mesmas configurações prévias do modo de análise de desempenho, com a diferença de que só é possível simular um algoritmo por vez para uma memória principal com uma única quantidade de frames.

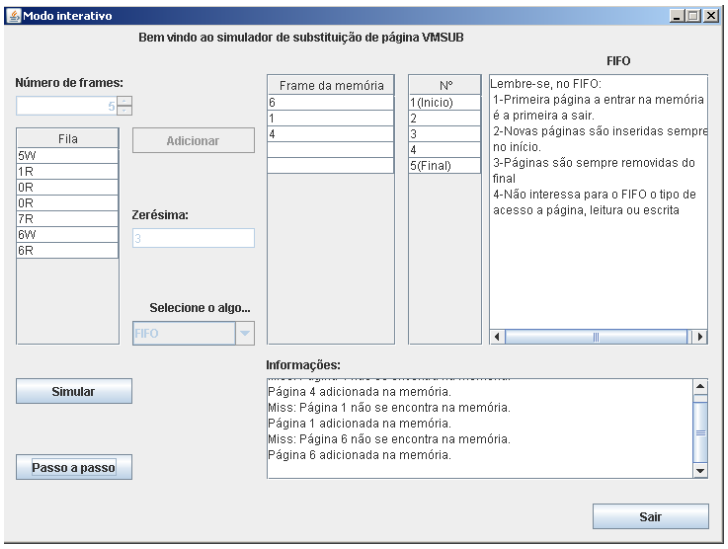


Figura 4. Modo interativo

Esse modo permite ao usuário simular manualmente o processo de substituição de páginas. O botão Adicionar, simula o carregamento de uma página por vez na memória, sendo visualizadas na fila de páginas. O botão ”passo a passo”, ao ser acionado, gera as informações de cada decisão a ser tomada no processo de substituição, mostrando os erros e acertos e qual página será substituída a cada *page fault*. Existem dois gráficos que representam as páginas nos *frames* da memória, permitindo uma melhor compreensão do processo de simulação. Além desses gráficos, há as informações sobre o algoritmo escolhido, que informam ao usuário como o algoritmo funciona e quais os critérios levados em consideração na substituição de cada página. Dessa forma,o usuário pode compreender, de forma prática, o funcionamento de cada algoritmo de substituição.

5. Resultados e discussões

Para medir a eficácia da ferramenta desenvolvida, ela foi apresentada a alguns alunos da disciplina de Sistemas Operacionais dos cursos de Engenharia de Telecomunicações e

Engenharia de Computação do IFCE, em uma espécie de aula experimental. Os alunos que participaram já tinham estudado, na disciplina, os conceitos de gerenciamento de memória virtual e o processo de substituição de páginas, incluindo alguns dos algoritmos. Após a aula experimental, utilizando a ferramenta, foi aplicado um questionário aos alunos, descrito na Tabela 2.

Questão 1	Questão 2	Questão 3	Questão 4	Questão 5
Você acha interessante esse tipo de Aplicação no ensino da disciplina?	**Quantos desses algoritmos de Substituição de Páginas você conhecia?**	**Com qual intensidade a aplicação foi útil nas opções abaixo? (1-Muito, 2-Parcialmente, 3-Insuficiente)**	**Você recomendaria esse software aos professores das disciplinas de S.O e afins?**	**Quais aspectos você acha que poderia melhorar? (É possível marcar mais de uma opção)**
a) Sim, pois facilita bastante o aprendizado e torna a aula mais dinâmica.	a) Todos.	a) Serviu para conhecer novos algoritmos de substituição de páginas.	a) Com certeza.	a) Interface gráfica.
b) Sim, ajuda um pouco, porém ainda deixa a desejar.	b) Mais de 5.	b) Ajudou-me a compreender o funcionamento prático dos algoritmos.	b) Sim, talvez poderia ajudá-lo.	b) Didática.
c) Muito pouco, não acredito que aplicações desse tipo na sala de aula possam influenciar tanto na minha aprendizagem.	c) Mais de 2.	c) A simulação serviu para comparar bem o desempenho dos algoritmos.	c) Não, não gostei da aplicação.	c) Poderia ter mais algoritmos.
d) Não acho interessante.	d) Um ou mais.			d) Acho que a aplicação está satisfatória.

Tabela 2. Questionário

O questionário foi elaborado de forma a obter a opinião dos alunos sobre a ideia de usar ferramentas que tornam a aula mais prática e buscar uma avaliação do aspecto e funcionalidade da aplicação em geral. A pesquisa foi realizada com 13 alunos dos cursos de Engenharia de Telecomunicações e Engenharia de Computação do IFCE. Os resultados obtidos do questionário estão descritos na Tabela 3.

Itens/Questões	Q1	Q2	Q3	Q4	Q5
a)	84,6%	16,7%	1)30,8%; 2)38,5%; 3)30,8%;	69,2%	Recebeu 9 votos
b)	7,7%	33,3%	1)30,8%; 2)38,5%; 3)30,8%;	30,8%	Recebeu 8 votos
c)	7,7%	33,3%	1) 38,5% 2) 15,4%; 3)46,2%;		Recebeu 3 votos
d)		16,7%			

Tabela 3. Respostas das questões

Como pode ser observado na Tabela 3, a maioria dos alunos considera importante o uso de ferramentas como a VMSub na sala de aula e 69,2% deles recomendariam a aplicação aos professores. Pode-se perceber, ainda, que apenas 16,7% dos alunos conheciam todos os algoritmos implementados. Nesse caso, a aplicação também foi útil, pois apresentou aos alunos novos algoritmos ainda não conhecidos por eles.

Já pelas estatísticas de resposta da questão 3, pode-se concluir que os alunos não conseguiram visualizar bem o desempenho de cada algoritmo. Isso aconteceu, provavelmente, porque os alunos não acharam tão satisfatória a interface gráfica da aplicação, já que 9 de 13 alunos acham que a interface gráfica pode melhorar.

6. Trabalhos Relacionados

Nesta seção são apresentados alguns trabalhos semelhantes a aplicação VMSub, visando comparará-los.

A Page Replacement Algorithm Simulation-SamSol **(SAMSOL, 2015)**: Desenvolvido pelo engenheiro de software Samir Solanki. Este software foi feito na linguagem java e simula dois algoritmos , o FIFO e o LRU.

PRA (*Page Replacement Algorithm* (PRA, 2015)): Desenvolvido por Ting Yu do Departamento de Ciências da Computação de Illinois usando a plataforma web simulando os algoritmos Ótimo, LRU e FIFO.

SimulaRSO (SIMULARSO, 2015): Desenvolvido pela Universidade Católica de Santos pelo orientador André Luiz Vizine Pereira e os alunos André de Araújo Rodrigues e Caio Ribeiro Pereira do curso de Sistemas de Informação-2011. Desenvolvido também para plataforma web. É capaz de simular algoritmos de escalonamento de disco, escalonamento de processos e paginação de memória virtual. Em relação a paginação de memória virtual é capaz de simular os algoritmos FIFO, LRU,Ótimo e MRU.

SoSIM (SOSIM, 2015): O SOsim foi desenvolvido pelo prof. Luiz Paulo Maia como parte de sua tese de mestrado no Núcleo de Computação Eletrônica da Universidade Federal do Rio de Janeiro (NCE/UFRJ).O objetivo deste trabalho foi desenvolver uma ferramenta gratuita que permitisse facilitar e melhorar as aulas de sistemas operacionais para alunos e professores. O simulador permite visualizar os conceitos de multiprogramação, processo e suas mudanças de estado, gerência do processador (escalonamento) e a gerência memória virtual.

A Tabela 4 faz uma comparação entre o VMSub e os trabalhos já citados. A análise é feita quanto ao número de algoritmos implementados por cada aplicação, plataforma utilizada e quais funções estão presentes na aplicação.

		VMsub	P.R.A	SamSol	SimulaRSO	SoSIM
Algoritmos simulados	FIFO	X	X	X	X	X
	Segunda Chance	X				
	LFU	X				
	MFU	X				
	Rand	X				
	MRU	X			X	
	NUR	X				
	LRU		X	X	X	
	Ótimo	X	X		X	
Plataforma	Java	X		X		
	Web		X		X	
	Delphi					X
Funções	Exibir resultado	X	X	X	X	X
	Passo-a-passo	X	X			
	Animação		X		X	X

Tabela 4. Comparativo com outras aplicações

No que diz respeito ao número de algoritmos implementados, a aplicação VMSub está em vantagem em relação as outras, pois implementa oito algoritmos, enquanto que as outras implementam, no máximo, quatro. Em relação à plataforma, as aplicações P.R.A e SimulaRSO possuem a vantagem de ser Web, o que as torna mais portáteis, facilitando o acesso dos alunos às mesmas, já que nada precisa ser instalado. Quanto às funções, a VMSub além de exibir os resultados das simulações dos algoritmos, mostra o ”passo a passo”da execução, visando facilitar o aprendizado do aluno. Apesar disso, a aplicação P.R.A se sobressai, pois além dessas funções, possui uma boa animação. A aplicações SoSIM, SamSol e SimulaRSO se destacam quanto a animação, mas não implementa o ”passo a passo”.

7. Conclusão e Trabalhos Futuros

Após a validação da aplicação, pode-se comprovar a importância do uso de ferramentas que tornem o aprendizado mais interativo e facilitem a visualização de forma prática, já que a maioria dos alunos disseram que esse tipo de ferramenta pode ajudá-los na sala de aula e que recomendariam aos professores. Apesar de algumas melhorias que precisam ser feitas na VMSub, a ferramenta foi útil de alguma maneira aos alunos, servindo para que pudessem aprender novos algoritmos de substituição e tornando possível visualizar o funcionamento prático dos mesmos.

Como trabalhos futuros, pretende-se melhorar a interface gráfica da aplicação, tornando mais fácil a compreensão, de forma gráfica, do desempenho de cada algoritmo. Pretende-se também implementar uma versão Web da aplicação, tornando-a mais portável e de fácil acesso. Após essas melhorias, a equipe desenvolvedora pretende propor aos professores das disciplina de Sistemas Operacionais do IFCE o uso da ferramenta em suas aulas.

Referências

DEITEL, H. M.; DEITEL, P. J.; CHOFFNES, D. R. **Sistemas Operacionais: terceira edição.** Pearson, 2005.

FONSECA, F. N.; NASCIMENTO, F. M. S. Uma Ferramenta de Simulacão do Processo de Substituicão de Páginas em Gerência de Memória Virtual. In: **XVII WORKSHOP SOBRE EDUCAÇÃO EM COMPUTAÇÃO (WEI)**, 2009, Bento Gonçalves -RS.

JÚNIOR, P. J. **Notas sobre Sistemas Operacionais.** Prentice Hall, 2004.

MAIA, L.; MACHADO, F. B. Um framework construtivista no aprendizado de sistemas operacionais - uma proposta pedagógica com o uso do simulador sosim. In: **XII Workshop Sobre Educação em Computação (WEI)**, 2004, Salvador - BA.

MAZIERO, C. A. **Sistemas Operacionais: Conceitos e Mecanismos.** Prentice Hall, 2014.

PRA, PAGE REPLACEMENT ALGORITHM. Disponível em: <http://goo.gl/42b2ky>. Acesso em: 15 jul. 2015.

SAMSOL. Disponível em: <http://goo.gl/73Qj73>. Acesso em: 15 jul. 2015.

SIMULARSO. Disponível em: <https://github.com/caio-ribeiro-pereira/SimulaRSO>. Acesso em: 15 jul. 2015.

SOSIM, SIMULADOR PARA O ENSINO DE SISTEMAS OPERACIONAIS VERSÃO 2.0.Disponível em: <http://www.training.com.br/sosim/>. Acesso em: 15 jul. 2015.

SÍLBERSCHATZ, A.; GALVIN, P.;GAGNE, G. **Sistemas Operacionais.** Campus, 2001.

TANENBAUM, A. S. **Sistemas Operacionais Modernos.** Pearson, 2010.